To create a console-based application for movie reviews using Java and MySQL with the goal of achieving maximum output, you need to implement all required functionalities efficiently and correctly. Maximum output likely means completing every feature specified—user authentication, profile management, movie display, review management, and review sharing—while ensuring the application is robust and meets all constraints. Below are step-by-step instructions to guide you through the process without providing code, focusing on a clear and systematic approach.

Step 1: Understand the Requirements

Before you begin, clarify what the application must do. The key functionalities include:

- **User Authentication**: Allow users to sign up and sign in.
- **Profile Management**: Enable users to edit their username and change their password.
- **Movie Management**: Display a fixed list of 10 movies.
- **Review Management**: Let users create, edit, delete, and view their reviews.
- **Review Sharing**: Allow users to share their reviews with others and view reviews shared with them.

Additional constraints to keep in mind:

- User and review IDs should be auto-generated.
- Users can only edit, delete, or share their own reviews.
- Users cannot share reviews with themselves.
- Deleting a review removes it from shares.
- Input validation is essential (e.g., no empty fields, valid movie selections).

Take time to list these requirements and constraints to ensure nothing is missed during development.

Step 2: Plan the Database Schema

Design a MySQL database to store all necessary data with proper relationships and constraints.

1. **Create a Database**:
    - Name it something meaningful, like movie_reviews_db.
2. **Define the Tables**:
    - **Users Table**: Stores user information.
        - Columns: user_id (auto-incremented unique ID), username (must be unique), password (store securely).
    - **Movies Table**: Holds the fixed list of 10 movies.
        - Columns: movie_id (auto-incremented unique ID), movie_name.
    - **Reviews Table**: Links users to movies with their reviews.
        - Columns: review_id (auto-incremented unique ID), movie_id (links to movies), user_id (links to users), review_text.

   - **Shares Table**: Tracks which reviews are shared with which users.
      - Columns: share_id (auto-incremented unique ID), review_id (links to reviews), shared_with_user_id (links to users).
3. **Set Up Relationships and Constraints**:
   - Use foreign keys to connect movie_id and user_id in the reviews table to their respective tables.
   - Ensure username in the users table is unique.
   - Add a rule in the shares table to prevent duplicate shares of the same review to the same user.
4. **Insert Fixed Movies**:
   - Manually add 10 movie names (e.g., "The Matrix", "Inception") into the movies table after setting it up.

Step 3: Set Up Your Java Project

Organize your development environment for efficiency.

1. **Start a New Project**:
   - Use an IDE like IntelliJ IDEA or Eclipse to manage your work.
   - Ensure you have the MySQL JDBC driver to connect Java to your database.
2. **Plan the Structure**:
   - Divide your work into logical sections:
      - One section for database operations (e.g., adding or retrieving data).
      - Another for the main program logic (e.g., menus and user interaction).
      - A separate area for representing data (e.g., user or review details).

Step 4: Design the Application Flow

Plan how the user will interact with the application through the console.

1. **Main Menu**:
   - Options: Sign Up, Sign In, Exit.
   - This is the starting point before a user logs in.
2. **Signed-In Menu**:
   - Options: Edit Profile, Change Password, Display All Movies, Create a Review, Edit Review, Delete Review, Display All Reviews, Display Reviews Shared with Me, Share Review, Sign Out.
   - This menu appears after a successful sign-in.
3. **User Interaction**:
   - Use text prompts to guide the user (e.g., "Enter username:").
   - Accept input via the console and process it accordingly.

Step 5: Implement User Authentication

Start with the foundation of your application.

1. **Sign Up**:
    - Ask for a username and password.
    - Check if the username is already taken in the database.
    - If unique, securely store the username and password (consider hashing the password for safety).
    - Assign an auto-generated user_id.
2. **Sign In**:
    - Prompt for username and password.
    - Verify the credentials against the database.
    - If correct, proceed to the signed-in menu; if not, show an error and ask again.

Step 6: Add Profile Management

Allow users to update their information.

1. **Edit Profile**:
    - Let the signed-in user enter a new username.
    - Check that the new username isn't already in use by another user.
    - Update the database with the new username.
2. **Change Password**:
    - Ask for a new password.
    - Ensure it's not empty, then update it in the database (hash it if you're using hashing).

Step 7: Handle Movie Display

Make the list of movies accessible.

1. **Display All Movies**:
    - Retrieve all 10 movies from the movies table.
    - Show them in the console with their IDs (e.g., "1. The Matrix").

Step 8: Manage Reviews

Enable users to work with their reviews.

1. **Create a Review**:
    - Show the movie list and ask the user to pick a movie by ID.
    - Check if the movie ID is valid.
    - Ask for review text and save it in the reviews table with the user's ID and movie ID.
2. **Edit Review**:

   - Display the user’s reviews (with IDs).
   - Ask which review to edit by ID.
   - Verify the review belongs to the user, then allow them to enter new text and update it.

3. **Delete Review**:
   - Show the user’s reviews and ask which one to delete by ID.
   - Confirm it’s theirs, then remove it from the reviews table and any related entries in the shares table.
4. **Display All Reviews**:
   - Fetch and show all reviews written by the signed-in user.

Step 9: Implement Review Sharing

Add the ability to share and view shared reviews.

1. **Share Review**:
   - Display the user’s reviews and ask which one to share.
   - Ask for the username of the recipient.
   - Check that the recipient exists and isn’t the same as the current user.
   - Add an entry to the shares table linking the review to the recipient.
2. **Display Reviews Shared with Me**:
   - Look up the shares table for entries where the signed-in user is the recipient.
   - Retrieve and display those reviews.

Step 10: Add Input Validation

Ensure the application handles errors gracefully.

- Check for empty inputs (e.g., username, password, review text).
- Validate movie IDs when creating reviews.
- Prevent actions on reviews not owned by the user (e.g., editing or deleting someone else’s review).
- Show clear error messages (e.g., "Invalid movie ID" or "Username already taken").

Step 11: Build and Test Incrementally

Work on one feature at a time and test it before moving on.

1. **Test Authentication**:
   - Sign up with a new user, try a duplicate username, and sign in with correct/incorrect details.
2. **Test Profile Changes**:
   - Change the username and password, ensuring updates work and constraints are enforced.

3. **Test Movie Display**:
   - Verify all 10 movies appear correctly.
4. **Test Review Features**:
   - Create, edit, and delete reviews, ensuring ownership rules apply.
   - Check that deleted reviews disappear from shares.
5. **Test Sharing**:
   - Share a review, try sharing with yourself (should fail), and verify the recipient sees it.

Step 12: Finalize and Polish

Review your work to ensure maximum output.

- **Error Handling**: Provide helpful feedback for all errors.
- **Security**: Store passwords securely (e.g., hashed, not plain text).
- **Testing**: Re-test every feature to confirm it meets the requirements.
- **Completeness**: Double-check that all listed functionalities (Sign In, Sign Up, etc.) are fully implemented.

Key Tips for Maximum Output

- **Focus on Completeness**: Since evaluation may depend on the number of working features, ensure every specified function is included.
- **Keep It Simple**: Stick to basic console input/output and straightforward database queries.
- **Stay Organized**: Break tasks into small, manageable steps to avoid confusion.
- **Test Thoroughly**: Catch and fix issues early to avoid last-minute problems.